Software Requirements Specification

for

RANCANG BANGUN APLIKASI PENILAIAN ESAI SINGKAT BERBAHASA INDONESIA DAN INGGRIS MENGGUNAKAN METODE TEST-DRIVEN DEVELOPMENT

Version 1.0 approved

Prepared by Geizka Rozilia Ruicosta (119140114)

Institut Teknologi Sumatera

2023

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

4. System Features 4

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

5. Other Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

6. Other Requirements 5

Appendix A: Glossary 5

Appendix B: Analysis Models 5

Appendix C: To Be Determined List 6

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

Dokumen ini dibuat untuk melengkapi aplikasi yang dibangun pada penelitian Tugas Akhir milik pengembang yang berjudul RANCANG BANGUN APLIKASI PENILAIAN ESAI SINGKAT BERBAHASA INDONESIA DAN INGGRIS MENGGUNAKAN METODE TEST-DRIVEN DEVELOPMENT dengan nama aplikasi Easysay dan membangun sebuah aplikasi web yang dapat mempermudah para pengajar dalam memeriksa jawaban esai pelajar.

## Document Conventions

Dokumen ini menggunakan istilah sebagai berikut.

|  |  |
| --- | --- |
| FE | FrontEnd |
| BE | BackEnd |
| Pengajar | Dosen / Guru |
| Pelajar | Siswa / Mahasiswa |
| ML | Machine Learning |

## Intended Audience and Reading Suggestions

Dokumen ini ditunjukkan kepada pengembang yang hendak meneruskan penelitian terkait aplikasi Easysay kedepannya, para guru, para dosen, dosen pembimbing, dan juga dosen penguji.

## Product Scope

Tujuan dari dibangunnya aplikasi Easysay ini adalah untuk mempermudah proses penilaian esai berbahasa Indonesia dan Inggris, khususnya pada kondisi penilaian banyak esai dimana satu pengajar perlu memeriksa banyak jawaban pelajar dalam satu waktu. Aplikasi ini dapat digunakan pada peramban web, baik melalui komputer maupun ponsel. Aplikasi melakukan penilaian dengan membandingkan teks jawaban pengajar dengan teks lain yang merupakan jawaban pelajar, ataupun berkas CSV berisi kumpulan jawaban pelajar.

## References

* <https://github.com/Shercosta/easysay>
* 103.37.125.146:1234

# Overall Description

## Product Perspective

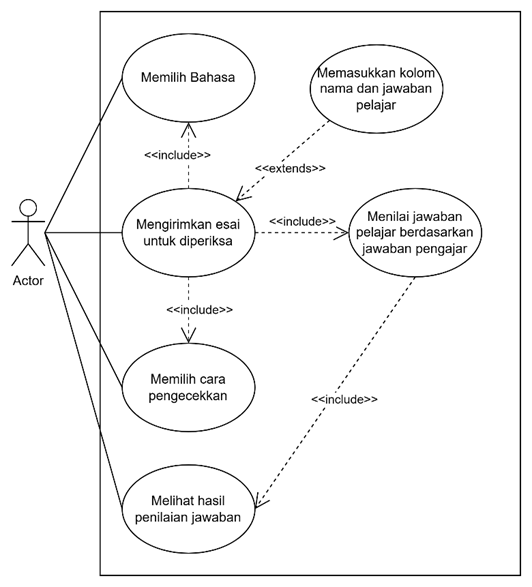
Aplikasi Easysay merupakan aplikasi yang akan dibuat menggunakan model ML berbahasa Inggris dan Indonesia yang sudah ada sebelumnya. Aplikasi akan dibuat untuk mengimplementasikan model ML yang telah dibuat tersebut.

## Product Functions

Aplikasi ini akan memiliki fungsi untuk dapat melakukan aktivitas sebagai berikut:

* Pengguna dapat memilih bahasa esai yang digunakan
* Pengguna dapat melakukan pemeriksaan esai dengan metode tunggal
* Pengguna dapat melakukan pemeriksaan esai dengan metode jamak
* Pengguna dapat melihat hasil penilaian esai

## User Classes and Characteristics

Pada aplikasi ini, hanya memiliki satu aktor, yaitu pengguna. Pada gambar dibawah ini dapat dilihat Use Case Diagram dari aplikasi yang akan dikembangkan.

## Operating Environment

Alat yang akan digunakan untuk mendukung pembuatan aplikasi ini adalah sebagai berikut.

1. *Notebook* dengan spesifikasi minimum, sistem operasi Windows 8, *processor* Intel Core i3 2330M CPU @ 2,2 GHz, memori 2GB DDR3, grafis NVIDIA GeForce GT 610 (4GB), *hardisk* 500GB. Pada tugas akhir ini digunakan komputer dengan spesifikasi minimum sistem operasi Windows 10, processor AMD Ryzen 5 2500U @ 2 GHz, memori 16GB DDR4, grafis AMD Radeon Vega 8, dan *Solid-State Drive* 512GB.
2. *Smartphone* dengan spesifikasi tipe minimum*,* OS Android OS v4.1.2 (Jelly Bean)*,* CPU Dual-core 800 MHz, GPU Mali-400, Internal 4 GB, 768 MB RAM*.* Pada tugas akhir ini digunakan OS Android 9.0 (Pie), CPU Octa-core 2.3GHz, GPU PowerVR GE8320, Internal 128 GB, 4GB RAM.
3. *Node.js* dengan versi minimal 16.16.0. Pada tugas akhir ini menggunakan *Node.js* dengan versi 18.12.1.
4. *Visual Studio Code* (VSCode) untuk menulis kode program menggunakan versi 1.74.2. Versi dari VSCode dapat berubah seiring adanya pembaruan dari pembangun aplikasi VSCode tersebut sewaktu-waktu.

## Design and Implementation Constraints

Batasan pada aplikasi yang akan dibuat ini dapat dilihat pada daftar berikut ini.

1. Aplikasi berbasis web hanya dapat digunakan melalui *browser*.
2. Aplikasi hanya dapat memproses penilaian esai yang diketik, bukan ditulis tangan.
3. Aplikasi hanya dapat menerima masukkan berupa teks untuk jawaban guru.
4. Aplikasi hanya dapat menerima masukkan berupa teks atau berkas *comma-separated values* (CSV) berisi kumpulan jawaban untuk jawaban murid.
5. Aplikasi dapat melakukan penilaian dengan teks minimal 1 kata hingga 100 kata.
6. Aplikasi hanya dapat melakukan penilaian terhadap 1 pertanyaan pada sekali pemakaian.
7. Penelitian ini terfokus pada implementasi aplikasi menggunakan model *machine learning* penilaian esai bahasa Indonesia dan bahasa Inggris yang sudah ada.

## User Documentation

Pengguna dapat melihat tutorial atau cara menggunakan aplikasi yang akan dibuat pada aplikasi yang akan dibuat saat sudah diluncurkan. Aplikasi akan diluncurkan dengan halaman bantuan guna mempermudah pengguna saat ingin menggunakan aplikasi.

## Assumptions and Dependencies

Apabila pengguna menggunakan aplikasi penilaian esai otomatis yang dikembangkan dan menggunakan fitur dibawah ini:

* Melakukan penilaian jamak menggunakan model bahasa Inggris
* Melakukan penilaian jamak menggunakan model bahasa Indonesia

Mengasumsikan bahwa pengguna telah menggunakan keduanya, pengembang mengembangkan cara menampilkan penilaian berbeda untuk kedua model. Hal ini dikarenakan dua model ini bekerja dengan cara berbeda, maka aplikasi yang dikembangkan juga menyesuaikan cara kerjanya berdasarkan model ML yang digunakan.

# External Interface Requirements

## User Interfaces

<Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

## Hardware Interfaces

<Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.>

## Software Interfaces

<Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.>

## Communications Interfaces

<Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.>

# System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## System Feature 1

<Don’t really say “System Feature 1.” State the feature name in just a few words.>

4.1.1 Description and Priority

<Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).>

4.1.2 Stimulus/Response Sequences

<List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.>

4.1.3 Functional Requirements

<Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use “TBD” as a placeholder to indicate when necessary information is not yet available.>

<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>

REQ-1:

REQ-2:

## System Feature 2 (and so on)

# Other Nonfunctional Requirements

## Performance Requirements

<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>

## Safety Requirements

<Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product’s design or use. Define any safety certifications that must be satisfied.>

## Security Requirements

<Specify any requirements regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.>

## Software Quality Attributes

<Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.>

## Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

# Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>